Final Exam

Programming Principles
(Spring 2016)

Choose and solve three problems among the following four problems via i-Campus. Each problem has its own entry in i-Campus. So, submit your solution to the entry that corresponds with the problem. Note that each problem is assigned a different score. Name your solution file in the format: "<student ID>-<problem code>.c". For example, "2015311999-A.c" or "2015311999-B.c".

A. Rope Crisis in Ropeland! (30 Points)

This is a story of Ropeland where rope pulling is a very popular game (like cricket in Bangladesh). Perhaps you know the game rope pulling: two groups of players hold two ends of a rope. When a certain signal is given they start pulling ropes. The group that can snatch the rope from the other group is declared winner. Today is a very happy day in Ropeland as they have got rope status (something like Bangladeshs test status). So the people of Ropeland are on the street and they are willing to be engaged in rope pulling. But the shops in the city fail to supply enough rope and so now a rope crisis has begun. The King of the country declares a new rule that two groups will not be allowed to buy more ropes than what they require.

The problem is that rope-pulling takes place in a large hall room that has a large round pillar in the middle with certain radius. So if two groups are on the opposite side of the pillar their pulled rope is never in a straight line. Given the position of the two groups you are to nd out the minimum length of rope required by them to start rope-pulling. You can assume that a point represents the position of each group.

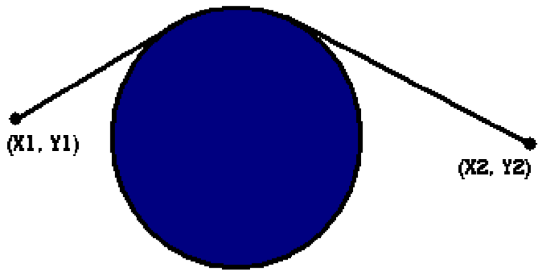

Fig 1: Situation when two groups have the round pillar between them. The pulled rope is never a straight line

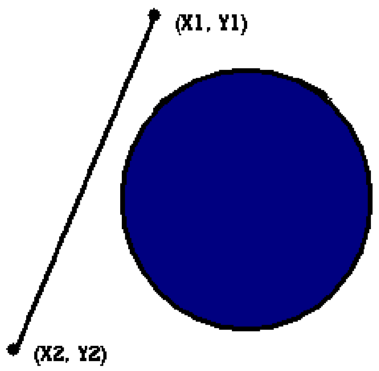

Fig 2: Situation when the two groups dont have the round pillar between them

Input Specification

The first line of the input contains an integer N, which tells how many sets of input are there. Next there are N lines of input. Each line contains five numbers X1, Y1, X2, Y2 and R

(> 0) where (X1, Y1) and (X2, Y2) are the coordinates of the two groups and R is the radius of the pillar. The coordinate of the center of the pillar is always the origin. You can also assume that none of the coordinates will be inside the circle. All input numbers except N are floating point numbers and none of their absolute value is greater than 10000.

Output Specification

For each set of input, output a floating-point number in a new line rounded to the third digit after the decimal point and this number denotes the minimum length of the rope required.

Example Input

```
2
1 1 -1 -1 1
1 1 -1 1 1
```

Example Output

```
3.571
2.000
```

B. Tug of War (40 Points)

A tug of war is to be arranged at the local office picnic. For the tug of war, the picnickers must be divided into two teams. Each person must be on one team or the other; the number of people on the two teams must not differ by more than 1; the total weight of the people on each team should be as nearly equal as possible.

Input Specification

The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs.

The first line of input contains n the number of people at the picnic. n lines follow. The first line gives the weight of person 1; the second the weight of person 2; and so on. Each weight is an integer between 1 and 450. There are at most 100 people at the picnic.

Output Specification

For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a blank line.
Your output will be a single line containing 2 numbers: the total weight of the people on one team, and the total weight of the people on the other team. If these numbers differ, give the lesser first.

Example Input

```
1
```

3
100
90
200

Example Output

190 200

C. Unlock the Lock (50 Points)

Mr. Ferdaus has created a special type of 4-digit lock named "FeruLock" shown in the picture on the left. It always shows a 4-digit value and has a specific unlock code (An integer value). The lock is unlocked only when the unlock code is displayed. This unlock code can be made to appear quickly with the help of some of the special buttons available with that lock. Each button has a number associated with it. When any of these buttons is pressed, the number associated with that button is added with the displayed value and so a new number is displayed. The lock always uses least significant 4 digits after addition. After creating such a lock, he has found that, it is also very difficult for him to unlock the Ferulock. As a very good friend of Ferdaus, your task is to create a program that will help him to unlock the Ferulock by pressing these buttons minimum number of times.

Input

There will be at most 100 test cases. For each test case, there will be 3 numbers: L, U and R where L ($0000 \leq L \leq 9999$) represents the current lock code, U ($0000 \leq U \leq 9999$) represents the unlock code and R ($1 \leq R \leq 10$) represents the number of available buttons. After that, there are R numbers ($0 \leq RV_i \leq 9999$) in a line representing the value of buttons. The values of L, U, RV_i will always be denoted by a four digit number (even if it is by padding with leading zeroes). Input will be terminated when L = U = R = 0.

Output

For each test case, there will be one line of output which represents the serial of output followed by the minimum number of button press required to unlock the lock. If it is not possible to unlock the lock, then print a line 'Permanently Locked' instead (without quotes).

Sample Input

```
0000 9999 1
1000
0000 9999 1
```

```
0001
5234 1212 3
1023 0101 0001
0 0 0
```

Sample Output

```
Case 1: Permanently Locked
Case 2: 9999
Case 3: 48
```

D. Boxes (60 Points)

We have some boxes numbered 1 to N. The dimensions of all boxes are identical. Now we have to stack up some of the boxes, subject to the following constraints:

1. One cannot put more than one boxes directly upon a box;
2. Boxes with lower serial numbers are not to be put upon one with a higher number;
3. The weight and maximum load for each box are given. The total weight of all boxes upon a box should not exceed its maximum load.

Please write a program that finds the maximum number of boxes that can be stacked up according to the above constraints.

Input

The first line of each set of input is an integer N ($1 \le N \le 1000$). This is followed by N lines, each with two integers, both ≤ 3000, representing the weight and maximum load of each box respectively.

Input ends with a case where $N = 0$.

Output

Each line of your output should give the number of boxes that can be stacked up.

Sample Input

```
5
19  15
7  13
5  7
6  8
1  2
0
```

Sample Output